Application Note AN3101-06: Four Channel 1-Bit D/A

by Pete Celi

Introduction

This paper describes an unconventional application for the DSP-1K. By directly writing to the serial output lines every 64 ticks, a 16X over-sampling 1-Bit D/A converter may be realized. The serial output pins provide analog information with a simple RC filter. The process involves calculating an interpolating FIR filter, then feeding those outputs into a Delta-Sigma modulator. Four channels may be implemented by using all four serial output pins. For this application, emphasis is on utility and not high performance.

Proposed Specs:

SNR: 54dB
BW: 15KHz
Passband Ripple: +/-.3dB
Stopband Rejection: -25dB
Over-sample Ratio: 16:1

Algorithm

Interpolation Filter

A 32-tap 1:8 interpolation filter is used to reduce the images present from the incoming 48KHz samples by 25dB. The 1:8 ratio means the filter operates as follows:

```
FILTER OUTPUT 0 = IN0*K0 + IN1*K8  + IN2*K16 + IN3*K24
FILTER OUTPUT 1 = IN0*K1 + IN1*K9  + IN2*K17 + IN3*K25
FILTER OUTPUT 2 = IN0*K2 + IN1*K10 + IN2*K18 + IN3*K26
FILTER OUTPUT 3 = IN0*K3 + IN1*K11 + IN2*K19 + IN3*K27
FILTER OUTPUT 4 = IN0*K4 + IN1*K12 + IN2*K20 + IN3*K28
FILTER OUTPUT 5 = IN0*K5 + IN1*K13 + IN2*K21 + IN3*K29
FILTER OUTPUT 6 = IN0*K6 + IN1*K14 + IN2*K22 + IN3*K30
FILTER OUTPUT 7 = IN0*K7 + IN1*K15 + IN2*K23 + IN3*K31
```

IN0 = most recently read 48KHz input sample
IN1 = input sample from previous sample period
IN2 = input sample from two sample periods previous
IN3 = input sample from three sample periods previous

Note: Each input is scaled by -2.0 as it is read in to preserve proper polarity and signal level for modulator algorithm.

The auto-decrementing address feature is on to allow easy FIR calculations for the 1:8 interpolator. The coefficients are as follows:

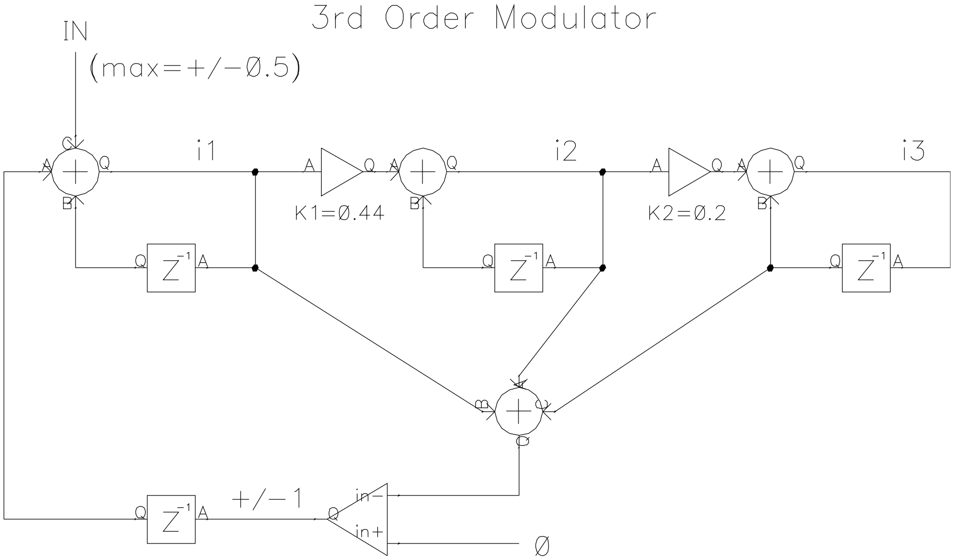
K0	=	3f7de4	=	K31
K1	=	3fc421	=	K30
K2	=	3fbfda	=	K29
K3	=	3fc364	=	K28
K4	=	3fd095	=	K27
K5	=	3fe88f	=	K26
K6	=	b87	=	K25
K7	=	38c4	=	K24
K8	=	6ea5	=	K23
K9	=	aa85	=	K22
K10	=	e8b0	=	K21
K11	=	12543	=	K20
K12	=	15c44	=	K19
K13	=	1895e	=	K18
K14	=	1a996	=	K17
K15	=	1ba4b	=	K16

Modulator

The interpolator filter outputs are converted into a 1-bit signal by a 3-stage Delta Sigma modulator. The modulator runs at 2x speed relative to the interpolator filter outputs, so the over-sample ratio is 2 x 8=16.

Modulator equations:

```
K1 = 0.44, K2 = 0.2
I1 = In + DAC + I1
I2 = K1*I1 + I2
I3 = K2*I2 + I3
Sum = I1 + I2 + I3
If (Sum>0) DAC = -1
Else DAC = 1
```



I1, I2, I3, and DAC are stored in the multi-purpose registers.

Source Code

In order for the 1-bit outputs to be interpreted correctly, each channel must be written to in exactly 64 tick increments. This may be accomplished by doing the first two channels of interpolation filter, followed by all four channels of the modulator routine, followed by the last two channels of interpolation filter, and then all four channels of the modulator routine again. This allows for exact intervals for writing to the four 1-bit outputs, and has the modulator running twice as fast as the interpolator.

The following shows a section of code to produce one interpolator filter output per channel and two 1-bit modulator outputs per channel.

;1K code for 4 channel 1-Bit D/A

```
;Interp CH0 out0
CM    0x380000          0x410 ;read in IN0*(-2) from CH0
SCA   0x3F7DE4          0x000 ; store input, A = IN0*K0
CMA   0x006EA5          0x001 ; A = A + IN1*K1
CMA   0x01BA4B          0x002 ; A = A + IN2*K2
CMA   0x0038C4          0x003 ; A = A + IN3*K3
SCA   0x000000          0x100 ; store output IntCH0 out0

;Interp CH1 out0
CM    0x380000          0x411 ; read in IN0*(-2) from CH1
SCA   0x3F7DE4          0x010 ; same routine as CH0
CMA   0x006EA5          0x011
CMA   0x01BA4B          0x012
CMA   0x0038C4          0x013
SCA   0x000000          0x110 ; store output IntCH1 out0

;Modulator 0
1MC   0x3C0000          0x400 ; A = DAC - 1 (sets A to +/-1)
CMA   0x040000          0x100 ; A = A + In (In = IntCH0 out0)
CMA   0x040000          0x401 ; A = A + I1
SCA   0x01CCCC          0x401 ; store new I1, A = I1*K1
CMA   0x040000          0x402 ; A = A + I2
SXCA  0x00CCCC          0x402 ; store new I2 into [M] and B
                       ; A = I2*K2
CMA   0x040000          0x403 ; A = A + I3
SCBA  0x040000          0x403 ; store new I3, A = I3 + I2
CMA   0x040000          0x401 ; A = A + I1 = I3 + I2 + I1
C    N  0x2000000             ; if A<0, A=2, else A=0
S1AC  0x3C0000          0x400 ; store A, A = A - 1, (sets A to +/-1)
C    !N 0xFFFFFFF      ; if A>=0, A = all bits high
SCA   0x000000          0x421 ; write A to output pin0, clear A

;Modulator 1, same idea as Modulator 0
1MC   0x3C0000          0x404
CMA   0x040000          0x110 ; input comes from IntCH1 out0
CMA   0x040000          0x405
SCA   0x01CCCC          0x405
CMA   0x040000          0x406
SXCA  0x00CCCC          0x406
CMA   0x040000          0x407
SCBA  0x040000          0x407
CMA   0x040000          0x405
C    N  0x2000000
S1AC  0x3C0000          0x404
C    !N 0xFFFFFFF
SCA   0x000000          0x422 ; write to  output pin1
```

```
;Modulator 2
1MC   0x3C0000          0x408
CMA   0x040000          0x121 ; input comes from IntCH2 out7
CMA   0x040000          0x409
SCA   0x01CCCC          0x409
CMA   0x040000          0x40A
SXCA  0x00CCCC          0x40A
CMA   0x040000          0x40B
SCBA  0x040000          0x40B
CMA   0x040000          0x409
C    N  0x2000000
S1AC  0x3C0000          0x408
C    !N 0xFFFFFFF
SCA   0x000000          0x424 ; write to output pin2

;Modulator 3
1MC   0x3C0000          0x40C
CMA   0x040000          0x131 ; input comes from IntCH3 out7
CMA   0x040000          0x40D
SCA   0x01CCCC          0x40D
CMA   0x040000          0x40E
SXCA  0x00CCCC          0x40E
CMA   0x040000          0x40F
SCBA  0x040000          0x40F
CMA   0x040000          0x40D
C    N  0x2000000
S1AC  0x3C0000          0x40C
C    !N 0xFFFFFFF
SCA   0x000000          0x428 ; write to output pin3

;Interp CH2 out0
CM    0x380000          0x412 ; read IN0*(-2) from CH2
SCA   0x3F7DE4          0x020
CMA   0x006EA5          0x021
CMA   0x01BA4B          0x022
CMA   0x0038C4          0x023
SCA   0x000000          0x120 ; store output IntCH2 out0

;Interp CH3 out0
CM    0x380000          0x413 ; read IN0*(-2) from CH3
SCA   0x3F7DE4          0x030
CMA   0x006EA5          0x031
CMA   0x01BA4B          0x032
CMA   0x0038C4          0x033
SCA   0x000000          0x130 ; store output IntCH3 out0

;Modulator 0
1MC   0x3C0000          0x400
CMA   0x040000          0x100 ; input from IntCH0 out0
CMA   0x040000          0x401
SCA   0x01CCCC          0x401
CMA   0x040000          0x402
SXCA  0x00CCCC          0x402
CMA   0x040000          0x403
SCBA  0x040000          0x403
CMA   0x040000          0x401
C    N  0x2000000
S1AC  0x3C0000          0x400
C    !N 0xFFFFFFF
SCA   0x000000          0x421
```

```
;Modulator 1
1MC   0x3C0000          0x404
CMA   0x040000          0x110 ; input from IntCH1 out0
CMA   0x040000          0x405
SCA   0x01CCCC          0x405
CMA   0x040000          0x406
SXCA  0x00CCCC          0x406
CMA   0x040000          0x407
SCBA  0x040000          0x407
CMA   0x040000          0x405
C    N  0x2000000
S1AC  0x3C0000          0x404
C    !N 0xFFFFFFF
SCA   0x000000          0x422

;Modulator 2
1MC   0x3C0000          0x408
CMA   0x040000          0x120 ; input from IntCH2 out0
CMA   0x040000          0x409
SCA   0x01CCCC          0x409
CMA   0x040000          0x40A
SXCA  0x00CCCC          0x40A
CMA   0x040000          0x40B
SCBA  0x040000          0x40B
CMA   0x040000          0x409
C    N  0x2000000
S1AC  0x3C0000          0x408
C    !N 0xFFFFFFF
SCA   0x000000          0x424

;Modulator 3
1MC   0x3C0000          0x40C
CMA   0x040000          0x130 ; input from IntCH3 out0
CMA   0x040000          0x40D
SCA   0x01CCCC          0x40D
CMA   0x040000          0x40E
SXCA  0x00CCCC          0x40E
CMA   0x040000          0x40F
SCBA  0x040000          0x40F
CMA   0x040000          0x40D
C    N  0x2000000
S1AC  0x3C0000          0x40C
C    !N 0xFFFFFFF
SCA   0x000000          0x428
```

This section of code is essentially repeated 8 times with new coefficients for the interpolator in each section.

Conclusion

This approach may be used to create a higher performance two-channel system. The key to enhancing noise performance is to use a higher order modulator. Increasing the number of coefficients in the interpolation filter will improve pass-band ripple and image rejection specs. An external return-to-zero (RTZ) circuit should be employed to condition the 1-bit output to obtain predicted results when trying to obtain higher performance.

NOTICE

Wavefront Semiconductor reserves the right to make changes to their products or to discontinue any product or service without notice. All products are sold subject to terms and conditions of sale supplied at the time of order acknowledgement. Wavefront Semiconductor assumes no responsibility for the use of any circuits described herein, conveys no license under any patent or other right, and makes no representation that the circuits are free of patent infringement. Information contained herein is only for illustration purposes and may vary depending upon a user's specific application. While the information in this publication has been carefully checked, no responsibility is assumed for inaccuracies.

Wavefront Semiconductor products are not designed for use in applications which involve potential risks of death, personal injury, or severe property or environmental damage or life support applications where the failure or malfunction of the product can reasonably be expected to cause failure of the life support system or to significantly affect its safety or effectiveness.

All trademarks and registered trademarks are property of their respective owners.

Contact Information:

Wavefront Semiconductor
200 Scenic View Drive
Cumberland, RI 02864 U.S.A.
Tel: +1 401 658-3670
Fax: +1 401 658-3680
On the web at www.wavefrontsemi.com
Email: info@wavefrontsemi.com

Application note revised March, 2005